Software Reuse and Reference Architecture Processes Study

Study Introduction

SEEDs Community Workshop #1

February 5-7, 2002

Version 2

Presenter: G. McConaughy

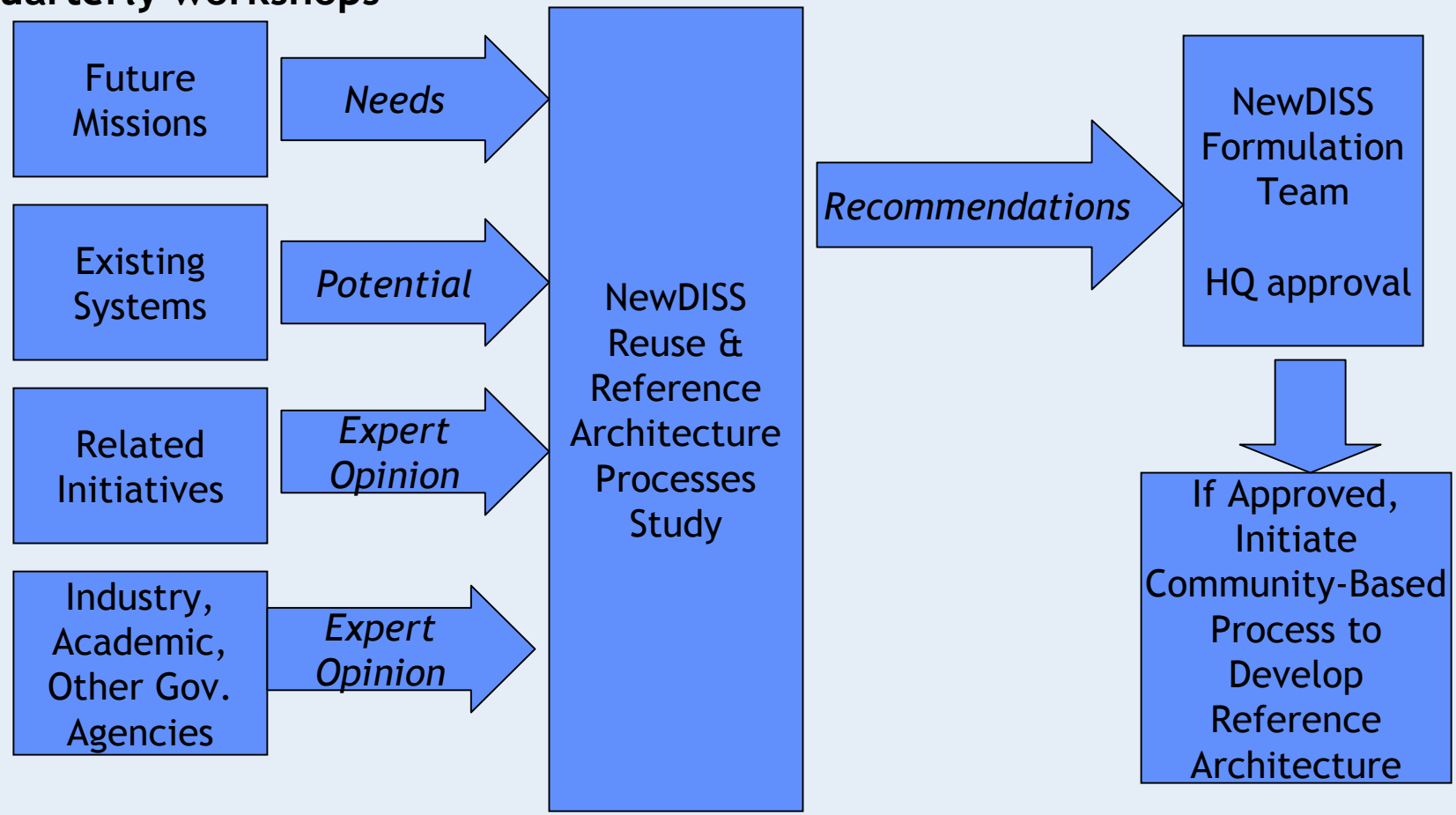
The Problem

- **Need a more cost effective DISS development approach for future missions**
  - Legacy systems may well consume most of the projected ESE information systems budget
  - "Expertise" & "smallness" large positive factor in cost effective development – leverage required
- **Need a more flexible/responsive development approach**
  - Very large development efforts require rigid requirements control
  - "Smaller" efforts respond more quickly
- **Need increased and effective/accountable community participation**
  - Centralized systems do not effectively leverage community expertise
  - Community systems may not effectively leverage each other or meet critical mission requirements (e.g., long-term data retention)

The Opportunity

- **Reuse and reference architectures can reduce system development costs**
  - Reuse can leverage large base of existing ESE software, system assets and expertise
  - Reused artifacts and components require less development and testing
  - Reference architectures can enable an efficient market of components and services
- **Reuse and reference architectures can improve flexibility & responsiveness**
  - Smaller development efforts can be effectively coordinated & integrated through the ref. Architecture
  - Assembly of new systems from reused or commodity components shortens schedules
- **Reference architectures can increase community participation**
  - Enables development to be performed wherever expert resources are available
  - Ensures interoperability of independently developed components & systems
  - Provides a clear demarcation for delivered functionality

- Reliance on stakeholder view of supply and demand – emphasis on practical experience of actual mission to mission reuse
- Key related initiatives examined for recommendations – e.g. Carnegie Mellon SEI, OGC, OMG, ETC.
- Feedback incorporated from ESE scientific community through interviews & quarterly workshops



Pre-work: Structure Analysis & Trades

- Initial interviews, review of documented case studies & internet material to date:
  - **Federation** NewDISS working group
  - **Related NASA initiatives**: Digital Earth Reference Model, Earth Science Modeling Framework, and the Information Power Grid, Renaissance, Open Archives Information System
  - **Current ESE systems**: ECS, TSDIS, SeaWiFS, ESIPS (Cornillon, …), DAACs (JPL, GSFC, ..), OMI, CEOS, GCMD, DIAL
  - **Future mission science systems**: Global Precipitation Mission, Total Column Ozone
  - **Related consortia:** OGC, FGDC, OMG, ISO,and CCSDS
  - **Software engineering groups**: CMU Software Engineering Institute, GSFC Software Engineering Laboratory
  - **Architecture framework initiatives**: Federal Enterprise Architecture Framework, C4ISR Architecture Framework, and the Zachman Framework, Weapons Systems Technical Architecture Working Group
  - **Government organizations** facing similar challenges: NIMA, NRO
  - **Industry Efforts:** McDonald Detweiler, NEC, GTE, Toshiba, DEC, HP, Raytheon, Fujitsu, Motorola

Results of Pre-Work: Range of Options/Evaluation Criteria

- **Evaluation Criteria Framework**
  - Cost savings over time
  - Increased Flexibility/Responsiveness to New Missions, Science, and Applications
  - Increase community participation
  - Assure effective and accountable participation
  - Suitability for NASA Future Consensus-Based Cultures
    - at least 2 identified: 1) cost & schedule-driven, 2) innovation-driven
- **Range of Options Identified** from community survey (e.g. CMU SEI, Linux Open Source, TSDIS/SeaWifs successes, Trends in Industry)

- **Solicit & Compile Community Views Formally (Workshops & Further Interviews)**
  - Gather community views from participants of this workshop
  - Gather additional community views after this workshop
  - Publish the community viewpoint toward software reuse and reference architectures
- **Interim Decision: Is there enough chance of success and community buy-in to proceed?**
  - If no: Stop the study and recommend no NASA new investment
  - If yes: Proceed to examine community-based processes to implement
- **Examine Community-Based Processes**
  - Interest in consensus-based processes done by actual stakeholders
  - Assure not one-size-fits-all – probably multiple working groups
  - Process is on-going, evolutionary – no big bang allowed
  - Interest in evolutionary test-bedding to prevent "systems-engineering-gone-mad" syndrome
  - Interest in leveraging work already done by other organizations if appropriate
- **Provide Trades, Analysis, Recommendations to HQ**
- **If HQ Approval & Funds Supplied: Initiate Community-Based Processes**
  - Charged with prioritization and implementation
  - Evolutionary

What can you expect to do at this workshop?

Two 1.5 hour sessions

- **Current trade space will be described (I.e. options & evaluation criteria)**
- **You will be asked to fill in your individual opinion on options using described evaluation criteria**
  - Since one-size-does-not-fit all we are seeking individual opinion, not group consensus. We will be analyzing responses to determine how many sizes might fit.
- **If you think we missed the boat, you are asked to provide new options and/or evaluation criteria with your rationale.**
  - We are happy to include new ideas that we might have missed.
  - This is a study. Now is the time to have open & constructive dialog about direction. Differences of viewpoint are not problems.

Backup Slides

Working Definitions

- **Reuse**
  - *Reuse* is the act of taking a functional capability used in (or provided by) one system or mission and employing it in another system or mission. This broad definition is intended to encompass a variety of techniques that have the potential to reduce future DISS costs, not simply libraries of reusable software components. For example, employing an entire existing system (including software, hardware, and operational processes) to support a new mission would fit this definition of reuse.
- **Architecture**
  - *Architecture* is defined as the structure of components, their interrelationships, and the principle guidelines governing their design and evolution over time (i.e., components, connections, and constraints).
  - A *reference architecture* is a generic architecture that provides coherent design principles for use in a particular domain (in this case, Earth science). It aims at structuring the design of specific system architectures by defining a unified terminology, a generic system structure, the kinds of system components, their responsibilities, dependencies, interfaces, data, behaviour (interactions), constraints, design rules, and models to represent all these aspects.
- **Product Line**
  - A *software product line* is a set of software-intensive systems sharing a common, managed set of features that satisfy the specific needs of a particular market segment or mission and that are developed from a common set of core assets in a prescribed way. (Carnegie Mellon SEI)

"Architecture" as used for the purposes of this study will be considered relative to how it supports integrating subsystems and/or components, where the underlying implementations of those components is assumed to be heterogeneous.

Backup

Software Reuse and Reference Architecture Processes Study

Questions to be Addressed:

Will actively promoting and investing in reuse and/or reference architectures provide the following return on investment?

1) Reduction of the cost of supporting future missions, science, and applications
2) Increased flexibility and responsiveness to new missions, science and applications
3) Increased effective, accountable community participation in system development and operations

If it will, what processes would best move ESE toward that goal? How can the community and NASA team to implement those processes?

- **Reuse Options & Evaluation Criteria Described (1/2 hour)**
  - Mark Nestler, GST
- **Reference Architecture Options & Evaluation Criteria Described (1/2 hour)**
  - David Isaac, BPS
- **"Worksheets" Completed by Community Representatives (1/2 hour)**
  - Roving Q&A Support
  - Reuse: Mark Nestler & Nadine Alameh (GST)
  - Reference Architectures: David Isaac (BPS)
- **Completed Worksheets Handed in at End of Session**
  - Worksheets with no name will be discarded
    - We will compile and publish opinions in the "aggregate" only – individual opinions will not be published and will be kept private
    - Your individual opinion is desired, group consensus is not required. Trying to avoid a one-size fits all approach.
  - Please "decline to comment" if you so choose (this also gives us some information)

Strategic Evolution of ESE Data Systems (SEEDS) Public Workshop
February 5-7, 2002
University of Maryland Inn & Conference Center

Software Reuse and Reference Architecture Process Study

Software Reuse and Reference Architecture Process Study: Workshop Goals

- **Increase awareness and understanding of stakeholders**
  - Software Reuse
    - What are the different approaches to software reuse?
    - How could software reuse benefit NASA ESE?
    - What are the costs and issues related to software reuse?
  - Reference Architecture
    - What is a reference architecture?
    - How could a reference architecture benefit NASA ESE?
    - What are the costs and issues related to developing a reference architecture?
- **Gather stakeholder input**
  - Does promoting software reuse seem worthwhile? Which approach might be best?
  - Does developing a reference architecture seem worthwhile? What level of detail might be appropriate?
  - Are there approaches to reuse or reference architectures other than those listed that NASA should consider?

Part 1: Software Reuse

Definitions

Evaluation Criteria

Alternatives: Reuse Options

Issues & Considerations

- **Reuse**
  - Taking a functional capability used in (or provided by) one system or mission and employing it in another system or mission
  - This functional capability can be in the form of code, or it can be design "artifacts" (e.g. architectures, software designs, ICDs, test plans, etc)
  - Broad definition for this study encompasses any means that avoids rebuilding a capability
  - Goals are to reduce costs, improve quality, increase productivity and speed system delivery, etc.

Evaluation Criteria

SELDS FOR LLULATION

- **Evaluate reuse options according to the following:**
  - Potential for Increasing System Cost Savings
    - Decreasing time-to-market
    - Improving development efficiency and productivity
    - Impact on system maintenance requirements
  - Potential for Increasing Flexibility and Responsiveness of Systems
    - Ability to accommodate new requirements
    - Ability to support new science applications
    - Ability to exploit new technologies
  - Potential for Increasing Effective and Accountable Community Participation
    - Ability to increase community participation
    - Ability to improve community accountability
  - Suitability for Flight Mission Needs
    - Fit of option with flight mission culture (cost & schedule)
    - Alignment of option organizational requirements with current organizational structure
  - Suitability for ESIP (and similar) Needs
    - Fit of option with ESIPs culture (innovation)
    - Alignment of option organizational requirements with current organizational structure
  - Investment Costs Required to Initiate and Support Process
    - Costs associated with process support/coordination costs
    - Costs associated with making existing assets reusable

- **Status Quo**
  - Continue employing current mix of practices including *ad hoc* "clone and own" and use of single centralized contractor
- **Improved "Clone & Own"**
  - Methodically copy existing assets (software & documents) and modify them as needed for use in a new system
  - Extends current practices with supporting processes and tools to allow it to be used more easily, by more groups, with better success
    - E.g., mechanisms for identifying, locating and understanding available systems, upgrading/creating documentation, providing easy access to system experts for extended consultation and guidance
- **Open Source Software Development**
  - Selected components/systems are collaboratively developed and updated by developers across missions
  - Repository control authority has final word on which upgrades/fixes/enhancements are incorporated into the base code
- **Encapsulated Services**
  - Wrap existing system or component with network-accessible interface, allowing access/use by others
- **Product Lines**
  - Structured, systematic (vs. opportunistic or *ad hoc*) reuse
  - Identify, create, maintain, and evolve common core assets that can be easily integrated to build sets ("lines") of related new systems ("products")

Some Considerations

- **Status Quo**
  - A varying record of timeliness within NASA ESE, depending upon process used (See next bullet)
- **Improved "Clone & Own"**
  - An ad-hoc form of this method is currently being practiced with success by some within ESE
  - Requires each cloned system to be maintained as a separate system
  - Technology upgrades/infusion can be problematic (risk of cloning technically obsolete systems)
- **Open Source Software Development**
  - Independent peer review and problem debugging often leads to better product quality
  - Reliability of components for various mission purposes is not guaranteed because these components are built by a variety of developers and for a variety of purposes
  - Long-term maintenance can be challenging because developers may lose interest in maintaining their component(s), or may be hesitant to support change requests from other groups
- **Encapsulated Services**
  - Organization that owns the wrapped component becomes a service provider, which could require a cultural shift in the organization
  - Maintenance of supplied service: organization that owns the wrapped component experiences net resource drains unless service "customers" help pay for operation
- **Product Lines**
  - High initial investment to build core assets requires at least two reuses to realize cost savings
    - Components must be more general to accommodate different "products"
    - Variation points must be built into architecture and core assets to allow for ease of tailoring of product lines
  - Requires new organizational structure that is not mission-driven
  - Advocated by CMU Software Engineering Institute

Part 2: Reference Architectures

Definitions

Evaluation Criteria

Alternative Approaches

Issues & Considerations

- **Architecture = High Level Design**
  - Components- the key functional pieces of a system
  - Connections- the relationships among components
  - Constraints- guidelines and rules for design and system evolution
- **Reference Architecture**
  - A generic architecture for use in a particular domain (e.g., Earth science)
  - Used as a reference when developing a specific system architecture
  - Goal is to have a common reference to promote component reuse, reduce integration costs, promote interoperability, etc.
    - *But our main focus is to examine the use of a reference architecture to enable reuse of software (code) and software development artifacts*
  - Could be high level or detailed
  - Focus on enabling application (domain-specific, vs. infrastructure) software reuse and application system openness

Evaluation Criteria

SELECTION

- **Evaluate each option according to the following:**
  - Potential for Cost Savings
    - Reducing solicitation and proposal effort
    - Increasing competition
    - Increasing development efficiency
    - Reducing integration effort
  - Potential for Increasing Flexibility and Responsiveness
    - Increasing responsiveness to new requirements
    - Improving support for new science activities/requirements
    - Supporting use of new technologies
  - Potential for Increasing Effective and Accountable Community Participation
    - Facilitating increased community participation
    - Facilitating community accountability
  - Suitability for Flight Mission Needs
    - Fit with flight mission culture (cost & schedule emphasis)
    - Alignment of approach's organizational requirements with current organizational structure
  - Suitability for ESIP (and similar) Needs
    - Fit with ESIP culture (innovation)
    - Alignment of approach's organizational requirements with current organizational structure
  - Investment Costs Required
    - Process support & coordination costs
    - Technical and documentation effort
    - Information dissemination costs

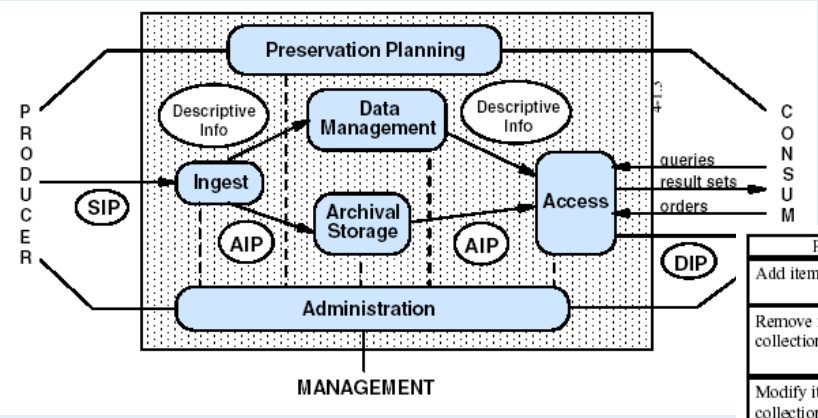
Alternatives Overview

- **How specific should a reference architecture be to be useful?**
  - Whatever we're doing now (status quo)
  - Notional
  - Concrete
  - Specific
- **How granular should the reference architecture components be?**
  - Coarse
  - Medium
  - Fine

Alternatives: Specificity

- **Status Quo**
  - NASA would continue involvement in related activities at current levels
- **Notional**
  - Defines subsystems/components and allocates requirements/functionality to each
  - Identifies key data flows
  - Examples: OpenGIS Abstract Specification Topic 12: OpenGIS Service Architecture; Reference Model for an Open Archive Information System; USIGS Objective System Architecture, OSI Reference Model
- **Concrete**
  - Identifies the services (including key parameters) of each subsystem/component in lay terms
  - Identifies all major data flows
  - Examples: OpenGIS Abstract Specification Topic 13: Catalog Services; USIGS Operational Architecture; TCP/IP Tutorial (RFC 1180)
- **Specific**
  - Defines the services (including all parameters) of each component in precise enough terms to build interfaces; defines the service invocation mechanism (call, post, get, etc.)
  - Identifies all data flows and specifies their format
  - Examples: OpenGIS Web Map Server Implementation Specification; USIGS Technical Architecture; TCP/IP standards suite (several dozen RFCs)

Alternatives: Specificity Examples



Notional

Function	Function Description	Function Inputs	Function Outputs
Add item to collection	Add new item to collection, at end of current collection list	item object	(none)
Remove item from collection	Remove one specified item from current collection list, and destroy that item object	item object reference, or partial item properties name and value list	(none)
Modify item in collection (optional)	Modify contents of one specified item in current collection list	item object reference, modified item properties name and value list	(none)
Copy selected item from collection	Create and return exact copy of one specified item object from current collection list	item object reference, or partial item properties name and value list	item object
Create iterator through collection (optional)	Create and return iterator object for all items in current collection list	(none)	new iterator object
Copy collection object (optional)	Create and return exact copy of this entire collection object, including all contained item objects	(none)	new collection object
	Get schema of specified item in current collection list, including definitions of all data elements in that item and all functions supported by that item object	item object reference, data element names list (optional)	data dictionary name and value list
rties	Retrieve specified properties of this collection object	collection properties names list	collection properties name and value list
rties	Modify specified properties of this collection object	collection properties name and value list	(none)
ject	Create new collection object of selected type. A separate factory object can be used if needed.	initial collection items list (optional), collection properties name and value list (optional)	new collection object
bject	Delete this collection object	(none)	(none)

Concrete

Table 7 — The Parameters of a GetMap Request

Request Parameter	Required/ Optional	Description
VERSION=version	R	Request version.
REQUEST=GetMap	R	Request name.
LAYERS=layer_list	R	Comma-separated list of one or more map layers. Optional if SLD parameter is present.
STYLES=style_list	R	Comma-separated list of one rendering style per requested layer. Optional if SLD parameter is present.
SRS=namespace:identifier	R	Spatial Reference System.
BBOX=xmin,ymin,xmax,ymax	R	Bounding box corners (lower left, upper right) in SRS units.
WIDTH=output_width	R	Width in pixels of map picture.
HEIGHT=output_height	R	Height in pixels of map picture.
FORMAT=output_format	R	Output format of map.
TRANSPARENT=TRUE|FALSE	O	Background transparency of map (default=FALSE).
BGCOLOR=color_value	O	Hexadecimal red-green-blue color value for the background color (default=0xFFFFFF).

Specific

SELDS FOLLOW-ON

Alternatives: Granularity

- **Coarse**
  - Defines external interfaces to major subsystems only
  - Use architecture primarily to promote community system interoperability and subsystem utilization
- **Medium**
  - Define key internal interfaces within major subsystems
  - Use architecture to promote functional component interoperability and reuse
- **Fine**
  - Define internal interfaces within applications or functional components
  - Use architecture to promote software module interoperability and reuse

- **Notional Reference Architecture**
  - Little detail means little investment to develop and easy to understand
  - Provides a common vocabulary to facilitate discussions
  - Does not have the specificity needed to facilitate subsystem/component integration
- **Concrete Reference Architecture**
  - Some detail means some investment to develop
  - Identifies specific services to provide a concrete understanding of requirements
  - Has enough detail to facilitate subsystem/component integration at the conceptual level…implementations will require wrappers or gateways
- **Specific Reference Architecture**
  - More detail means substantial investment to develop and more material to understand
  - Defines interfaces to provide specific expectations of each subsystem or component
  - Has the specificity needed to enable low-cost subsystem/component integration

Evaluation of Alternatives

Evaluator Information

Reuse Worksheet

Reference Architecture Worksheet

Evaluation of Alternatives

- **Using the "worksheets" provided, please give us your individual opinion on the options presented using described evaluation criteria**
  - Since one-size-does-not-fit all we are seeking individual opinion, not group consensus. We will be analyzing responses to determine how many sizes might fit.
- **If you think we missed the boat, please provide new options and/or evaluation criteria with your rationale.**
  - We are happy to include new ideas that we might have missed.
- **Roving Q&A support while you are completing the worksheets**
  - Reuse: Mark Nestler & Nadine Alameh (GST)
  - Reference Architectures: David Isaac (BPS)
- **Completed Worksheets Handed in at End of Session**
  - Worksheets with no name will be discarded
    - We will compile and publish opinions in the "aggregate" only – individual opinions will not be published and will be kept private
  - Please "decline to comment" if you so choose (this also gives us some information)

- **Please provide the following information**
  - Name: _____
  - Organization: _____
  - Current activity: _____
  - Discipline: _____
  - Experience base: _____
  - Primary focus (circle one): Data Analysis
    System Development
    Management
    Other (specify): _____
  - Email workshop results to: _____

Reuse Evaluation Worksheet

SEEDS Foundation

Criteria \ Approach	Status Quo	Improved Clone & Own	Open Source	Service Encapsulation	Product Lines
1. System cost savings	- / 0 / +	- / 0 / +	- / 0 / +	- / 0 / +	- / 0 / +
2. Flexibility & responsiveness	- / 0 / +	- / 0 / +	- / 0 / +	- / 0 / +	- / 0 / +
3. Increased effective & accountable community participation	- / 0 / +	- / 0 / +	- / 0 / +	- / 0 / +	- / 0 / +
4. Suitability for ESE					
4.1. Mission	- / 0 / +	- / 0 / +	- / 0 / +	- / 0 / +	- / 0 / +
4.2. Science/Apps	- / 0 / +	- / 0 / +	- / 0 / +	- / 0 / +	- / 0 / +
5. Investment cost	L / M / H	L / M / H	L / M / H	L / M / H	L / M / H

For criteria 1-4, rate each alternative negative (-), neutral (0), or positive (+) in terms of the potential benefit in each area (i.e., support for SEEDS goals).

For criterion 5, rate each alternative low (L), medium (M), or high (H) in terms of the expected funding required to realize the rated benefits in 1-4.

Provide key rationale and other comments on the following page.

Add new options or criteria as needed and explain on following page.

- Key Rationale & Comments

- Additional Alternatives and Evaluation Criteria

Reference Architecture Evaluation Worksheet

SEEDS FOUNDATION

Criteria \ Alternative	Status Quo	Notional	Concrete	Specific
1. System cost savings	- / 0 / +	- / 0 / +	- / 0 / +	- / 0 / +
2. Flexibility & responsiveness	- / 0 / +	- / 0 / +	- / 0 / +	- / 0 / +
3. Increased effective & accountable community participation	- / 0 / +	- / 0 / +	- / 0 / +	- / 0 / +
4. Suitability for ESE				
4.1. Mission	- / 0 / +	- / 0 / +	- / 0 / +	- / 0 / +
4.2. Science/Apps	- / 0 / +	- / 0 / +	- / 0 / +	- / 0 / +
5. Investment cost	L / M / H	L / M / H	L / M / H	L / M / H

Criteria \ Granularity	Coarse	Medium	Fine
1 System cost savings	- / 0 / +	- / 0 / +	- / 0 / +
2 Flexibility & responsiveness	- / 0 / +	- / 0 / +	- / 0 / +
3 Increased effective & accountable community participation	- / 0 / +	- / 0 / +	- / 0 / +
4 Suitability for ESE			
4.1 Mission	- / 0 / +	- / 0 / +	- / 0 / +
4.2 Science/Apps	- / 0 / +	- / 0 / +	- / 0 / +
5 Investment cost	L / M / H	L / M / H	L / M / H

For criteria 1-4, rate each alternative negative (-), neutral (0), or positive (+) in terms of the potential benefit in each area (i.e., support for SEEDS goals).

For criterion 5, rate each alternative low (L), medium (M), or high (H) in terms of the expected funding required to realize the rated benefits in 1-4.

Provide key rationale and other comments on the following page.

Add new options or criteria as needed and explain on following page.

- Key Rationale & Comments

- Additional Alternatives and Evaluation Criteria